CS61065 Theory and Applications of Blockchain

Assignment 5: Hyperledger Fabric DApp

Date: 1 October
Deadline: 22nd Oct 2021 ; 11:55 PM

In this assignment, we will get familiar with application development using Hyperledger Fabric. We will learn how to programmatically access a Fabric test network, and invoke and query chaincodes deployed in it.

Submission Instructions

- Make a folder named 'A5-ROLLNUMBER'. That folder should contain your application code, and the output file generated using the provided chaincode and the testcase with the name "`output.txt`".
- Make sure you include all the necessary files to build and run your code such as `package.json` / `go.mod`.
- If you are using NodeJS, name the entry point to your application as "`main.js`". Similarly, if you use Golang, name it as "`main.go`". Do not include libraries such as `node_modules` directory.
- Compress the folder 'A5-ROLLNUMBER.zip' and submit it on the moodle.

Problem Statement

In this assignment, you will work with a sample chaincode for managing property ownership in Hyperledger Fabric. The chaincode is available at https://github.com/vipray/CS61065-Assignment-5-Hyperledger-Fabric. You need to download the chaincode and deploy it in the test-network, and write a DApp - a Fabric Application to invoke and query the same.
The steps to deploy the chaincode in the test-network are as follows:

Start the test-network with Fabric CA enabled using the command:
`./network.sh up createChannel -ca`

Make sure you pass the option "-ca", this will start the Fabric network with Fabric CAs.
The default channel name will be used which is "mychannel".

Do not alter any docker compose files in the test-network.

Clone the chaincode using:
`git clone https://github.com/vipray/CS61065-Assignment-5-Hyperledger-Fabric.git`

Deploy the provided chaincode using the command:
`./network.sh deployCC -ccn fabhouse -ccp PathToChaincodeDirectory -ccl go`

This will deploy the chaincode with name "fabhouse" for which you need to write an application.

Chaincode Explanation:

The fabhouse chaincode is used for property management, letting participant organizations to register a house, and manage its ownership.
It stores the following information for a house:

- NagarPalikaID **(Key)**
- Address
- Owner
- Size

This Chaincode supports the following methods:

- **CreateHouse(ctx contractapi.TransactionContextInterface, id string, address string, size int) error**
  - It will return the error (if any)
  - This function takes 'id', 'address' and 'size' of the house as input.
  - The Owner of this house will be the organization that is executing this transaction. It will identify the organization executing this transaction using **ctx.GetClientIdentity().GetMSPID()** function. For example, if an Org1 client is making the transaction, then the owner will be "Org1MSP".
  - Then it checks(using **HouseExists()**) if a house with 'id' is already present in the ledger or not.
    - If present, it returns 'already present' as an error.
    - Otherwise, it adds 'id', 'address', 'owner' and 'size' of house entry on the ledge.

- **HouseExists(ctx contractapi.TransactionContextInterface, id string)(bool, error)**
  - This function takes 'id' as input.
  - It returns two values,
    - boolean
    - error (if any)
  - It returns true if it already exists, false otherwise.

- **ReadHouse(ctx contractapi.TransactionContextInterface, id string)(string, error)**

    - This function takes ‘id’ as input.
    - It returns two values,
        - the House’s data corresponding to ‘id’
        - error (if any)

- **GetAllHouses(ctx contractapi.TransactionContextInterface)([]*House, error)**
    - It returns two values,
        - all Houses data found in world state of ledger
        - error (if any)
    - Then it uses **ctx.GetStub().GetStateByRange()** to access the data of Houses.

- **TransferHouse(ctx contractapi.TransactionContextInterface, id string, newOwner string)(error)**
    - **It will return the error if the client which is trying to transfer this house doesn't belong to the organization which has the ownership of the house.**
    - In simple terms, a house can only be transferred by the organization which has its ownership.
    - This function will take ‘id’ and ‘newOwner’ as input.
    - Then it checks(using **ctx.GetClientIdentity().GetMSPID()**) if this client has the right to transfer this house to someone else or not.
        - If not, return ‘you are not the owner of this house’ as an error.
        - Otherwise, make ‘newOwner’ as the owner of the house corresponding to the ‘id’ and update this data into the ledger using **ctx.GetStub().PutState()** function. Note, the ‘newOwner’ argument can be any arbitrary string. If the new owner string does not match any current participant’s MSP Id, then it simply cannot be transferred to a new owner by anyone unless a new MSP Id with that identical string is registered in the network by a new participant.

Application

Write a Fabric application to invoke and query the given fabhouse chaincode. An application can interact with a blockchain network by submitting transactions to a ledger or querying ledger content. While chaincode is the smart contract that is executed in a decentralized manner through consensus, the Fabric application is the layer that uses the Fabric SDK to execute such smart contracts.

The application will take 3 arguments -
The first two arguments will be the absolute paths to the connection profile of org1 and org2. For the test network, the connection profile is usually found at:
test-network/organizations/peerOrganizations/org1.example.com/connection-org1.json
The third argument will be the absolute path to a testcase file.

In your application, first you need to perform the following for two orgs - **Org1 and Org2 both**:

- Load the connection configuration file for the orgs.
- Create separate wallets for each org.
- Enroll the admin users. In test network the Fabric CA starts with the default username:password as admin:adminpw
- Enrolls application users, one for each org.
- Create and setup gateways for each org.
- Build separate network instances for the two orgs.

After this, you need read the testcase file and invoke and query the following methods accordingly:
CreateHouse
ReadHouse
TransferHouse

NOTE: ReadHouse and GetAllHouses methods are to be queried, not invoked.

Test case Explanation:

A testcase file will be provided, which contains one testcase per line. Your application should take this testcase file as input and perform transactions according to the attributes. From the result of the transactions, you need to print the output exactly as specified next.

Ex-
org1; CreateHouse; ap-123; 2nd Floor, MMM hall, IIT Kharagpur; 2000

- First attribute(org1) is the name of the organization from which you are required to execute the transaction.

- Second attribute(CreateHouse) corresponds to the function name as per the chaincode that you need to query/invoke.

- Following attributes(if any) are arguments to the methods, will depend upon the function which is required to be called.

- All the attributes are separated with **a semicolon followed by a single space**
  i.e -> “; ” (without quotes)

Output Explanation:

For each testcase line, you have to print three things

- **First, the list of attributes of that testcase line.**
    - Ex- For testcase line:
        - **org1; CreateHouse; a1; Teen Murti Marg Area, New Delhi; 2000**
    - Output Should be:
        - **['org1', 'CreateHouse', 'a1', 'Teen Murti Marg Area, New Delhi', '2000']**
- **Second, the output you get(if any) after executing the transaction in the form of a string.**
  If there is an error then only print "ERROR".
    - Ex- For testcase line:
        - **org2; ReadHouse; a1**
    - Output Should be:
        - **{"NagarPalikaID":"a1","Owner":"Org1MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000}**

- **Third, a new line.**

Sample Testcase File:

org1; CreateHouse; a1; Teen Murti Marg Area, New Delhi; 2000
org2; ReadHouse; a1
org2; CreateHouse; b2; Mouseton, Calisota, U.S.; 200
org2; GetAllHouses
org1; CreateHouse; a2; 2nd Floor, MMM hall, IIT Kharagpur; 200
org1; GetAllHouses
org1; TransferHouse; a4; GOI
org2; CreateHouse; b1; Baker Street, London; 2000
org2; ReadHouse; 221B
org2; TransferHouse; a2; BegalGovt
org2; ReadHouse; b1
org1; ReadHouse; a5
org1; TransferHouse; a2; BegalGovt
org1; TransferHouse; a1; Org2MSP
org1; GetAllHouses
org1; TransferHouse; a2; Org2MSP
org1; GetAllHouses

org2; TransferHouse; a1; GOI
org2; GetAllHouses
org2; TransferHouse; b3; Org1MSP
org2; GetAllHouses
org2; TransferHouse; b1; Org1MSP
org2; GetAllHouses
org1; CreateHouse; a3; Malibu, California, United States; 1111
org2; CreateHouse; b3; Org2 Office, Org2 Road, Org2 City; 1111
org2; TransferHouse; a3; BegalGovt
org2; GetAllHouses
org1; GetAllHouses

Sample Output:

['org1',
  'CreateHouse',
  'a1',
  'Teen Murti Marg Area, New Delhi',
  '2000']

['org2', 'ReadHouse', 'a1']
{"NagarPalikaID":"a1","Owner":"Org1MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000}

['org2', 'CreateHouse', 'b2', 'Mouseton, Calisota, U.S.', '200']

['org2', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"Org1MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org1',
  'CreateHouse',
  'a2',

'2nd Floor, MMM hall, IIT Kharagpur',
'200']

['org1', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"Org1MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"Org1MSP","Address":"2nd Floor, MMM hall, IIT
Kharagpur","Size":200},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org1', 'TransferHouse', 'a4', 'GOI']
ERROR

['org2', 'CreateHouse', 'b1', 'Baker Street, London', '2000']

['org2', 'ReadHouse', '221B']
ERROR

['org2', 'TransferHouse', 'a2', 'BegalGovt']
ERROR

['org2', 'ReadHouse', 'b1']
{"NagarPalikaID":"b1","Owner":"Org2MSP","Address":"Baker Street, London","Size":2000}

['org1', 'ReadHouse', 'a5']
ERROR

['org1', 'TransferHouse', 'a2', 'BegalGovt']

['org1', 'TransferHouse', 'a1', 'Org2MSP']

['org1', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"Org2MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"b1","Owner":"Org2MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org1', 'TransferHouse', 'a2', 'Org2MSP']
ERROR

['org1', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"Org2MSP","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"b1","Owner":"Org2MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org2', 'TransferHouse', 'a1', 'GOI']

['org2', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"GOI","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"b1","Owner":"Org2MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org2', 'TransferHouse', 'b3', 'Org1MSP']
ERROR

['org2', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"GOI","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"b1","Owner":"Org2MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org2', 'TransferHouse', 'b1', 'Org1MSP']

['org2', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"GOI","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"b1","Owner":"Org1MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200}]

['org1',
  'CreateHouse',
  'a3',
  'Malibu, California, United States',
  '1111']

['org2',
  'CreateHouse',
  'b3',
  'Org2 Office, Org2 Road, Org2 City',
  '1111']

['org2', 'TransferHouse', 'a3', 'BegalGovt']
ERROR

['org2', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"GOI","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"a3","Owner":"Org1MSP","Address":"Malibu, California, United States","Size":1111},{"NagarPalikaID":"b1","Owner":"Org1MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200},{"NagarPalikaID":"b3","Owner":"Org2MSP","Address":"Org2 Office, Org2 Road, Org2 City","Size":1111}]

['org1', 'GetAllHouses']
[{"NagarPalikaID":"a1","Owner":"GOI","Address":"Teen Murti Marg Area, New Delhi","Size":2000},{"NagarPalikaID":"a2","Owner":"BegalGovt","Address":"2nd Floor, MMM hall, IIT Kharagpur","Size":200},{"NagarPalikaID":"a3","Owner":"Org1MSP","Address":"Malibu, California, United States","Size":1111},{"NagarPalikaID":"b1","Owner":"Org1MSP","Address":"Baker Street, London","Size":2000},{"NagarPalikaID":"b2","Owner":"Org2MSP","Address":"Mouseton, Calisota, U.S.","Size":200},{"NagarPalikaID":"b3","Owner":"Org2MSP","Address":"Org2 Office, Org2 Road, Org2 City","Size":1111}]